

PyPalEx

2.0.0

Generated by Doxygen 1.9.8

1 PyPalEx: The Python Palette Extractor	1
1.1 Description	1
2 Namespace Index	1
2.1 Package List	1
3 Class Index	2
3.1 Class List	2
4 File Index	2
4.1 File List	2
5 Namespace Documentation	3
5.1 pypalex Namespace Reference	3
5.1.1 Detailed Description	3
5.2 pypalex.__main__ Namespace Reference	3
5.2.1 Function Documentation	4
5.2.2 Variable Documentation	6
5.3 pypalex.arg_messages Namespace Reference	7
5.3.1 Function Documentation	8
5.4 pypalex.constants Namespace Reference	8
5.4.1 Variable Documentation	9
5.5 pypalex.conversion_utils Namespace Reference	13
5.5.1 Function Documentation	13
5.6 pypalex.extraction_utils Namespace Reference	15
5.6.1 Function Documentation	16
5.7 pypalex.Extractor Namespace Reference	22
5.8 pypalex.file_utils Namespace Reference	22
5.8.1 Function Documentation	22
5.9 pypalex.image_utils Namespace Reference	23
5.9.1 Function Documentation	23
5.10 pypalex.print_utils Namespace Reference	24
5.10.1 Function Documentation	25
6 Class Documentation	27
6.1 Extractor Class Reference	27
6.1.1 Detailed Description	28
6.1.2 Constructor & Destructor Documentation	28
6.1.3 Member Function Documentation	28
6.1.4 Member Data Documentation	31
7 File Documentation	33
7.1 __main__.py File Reference	33
7.1.1 Detailed Description	34

7.1.2 Author(s)	34
7.2 arg_messages.py File Reference	34
7.2.1 Detailed Description	35
7.2.2 Author(s)	35
7.3 constants.py File Reference	35
7.3.1 Detailed Description	36
7.3.2 Author(s)	36
7.4 conversion_utils.py File Reference	37
7.4.1 Detailed Description	37
7.4.2 Author(s)	37
7.5 extraction_utils.py File Reference	37
7.5.1 Detailed Description	38
7.5.2 Author(s)	38
7.6 Extractor.py File Reference	39
7.6.1 Detailed Description	39
7.6.2 Author(s)	39
7.7 file_utils.py File Reference	39
7.7.1 Detailed Description	40
7.7.2 Author(s)	40
7.8 image_utils.py File Reference	40
7.8.1 Detailed Description	40
7.8.2 Author(s)	40
7.9 print_utils.py File Reference	41
7.9.1 Detailed Description	41
7.9.2 Author(s)	41
Index	43

1 PayPalEx: The Python Palette Extractor

1.1 Description

PayPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

2 Namespace Index

2.1 Package List

Here are the packages with brief descriptions (if available):

pypalex	
Python Palette Extractor : extracts color palettes from images	3
pypalex.__main__	3
pypalex.arg_messages	7
pypalex.constants	8
pypalex.conversion_utils	13
pypalex.extraction_utils	15
pypalex.Extractor	22
pypalex.file_utils	22
pypalex.image_utils	23
pypalex.print_utils	24

3 Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Extractor	
Extracts colors given a matrix of HSV values extracted from an image	27

4 File Index

4.1 File List

Here is a list of all files with brief descriptions:

__main__.py	
Main script for PyPalEx	33
arg_messages.py	
Archive of messages to display for arguments supplied by user	34
constants.py	
A collection of constants for PyPalEx	35
conversion_utils.py	
Utilities for converting between RGB, HSV, HEX	37
extraction_utils.py	
Utilities for extracting colors from the image	37
Extractor.py	
Extraction utility class for extracting colors from the image	39

file_utils.py	Utilities for file handling	39
image_utils.py	Utilities for processing image and file handling	40
print_utils.py	Utilities for printing preview to the screen	41

5 Namespace Documentation

5.1 pypalex Namespace Reference

Python Palette [Extractor](#): extracts color palettes from images.

Namespaces

- namespace [__main__](#)
- namespace [arg_messages](#)
- namespace [constants](#)
- namespace [conversion_utils](#)
- namespace [extraction_utils](#)
- namespace [Extractor](#)
- namespace [file_utils](#)
- namespace [image_utils](#)
- namespace [print_utils](#)

5.1.1 Detailed Description

Python Palette [Extractor](#): extracts color palettes from images.

PyPalEx is a tool for extracting color palettes from images and generating a JSON format file with light and dark color themes. This tool is intended to be OS independent, for use by the tech community for developing their own custom theme managers or by artists who want to extract color palettes for their art from images, pictures or wallpapers they adore.

5.2 pypalex.__main__ Namespace Reference

Functions

- [main](#) ()
Main script function.
- [handle_args](#) ()
Handles the arguments passed to PyPalEx.
- [extract_color_palettes](#) ()
Handles color extraction from image(s).
- [setup_argument_parser](#) ()
Sets up the argument parser for command line arguments.
- [check_sources](#) (filepaths, path=None)
Checks each of the sources provided and removes any bad sources.
- [check_path](#) (path)
Check the path to make sure it exists.
- [set_global_args](#) (args)
Sets the global variables using the arguments.
- [check_source](#) (filepath)
Checks to make sure the path leads to a file.

Variables

- list `EXTRACTORS` = []
List of Extractor class objects for each individual image.
- list `PROPER_IMAGES` = []
List of real/existing image file path(s).
- list `FILENAMES` = []
List of image filenames.
- list `OUTPUT_FILEPATHS` = []
List of output file path(s) for each image.
- str `OUTPUT_PATH` = "
The path to the output directory where all JSON files will be saved.
- str `OUTPUT_TAIL` = "-color_palette.json"
The tail to append to each output filepath.
- bool `SAVE_CHECK` = False
Flag to check if user wants to save extracted color palettes.
- bool `SHOW_PREVIEW` = False
Flag to show a preview of extracted palettes.
- bool `PASTEL_L` = False
Flag to convert light color palette to pastel.
- bool `PASTEL_N` = False
Flag to convert normal color palette to pastel.
- bool `PASTEL_D` = False
Flag to convert dark color palette to pastel.

5.2.1 Function Documentation

`check_path()`

```
check_path (
    path )
```

Check the path to make sure it exists.

Parameters

<i>path</i>	The path to a directory.
-------------	--------------------------

Returns

True if the path exists and is not a file, False otherwise.

`check_source()`

```
check_source (
    filepath )
```

Checks to make sure the path leads to a file.

Parameters

<i>filepath</i>	Path to file with filename and file extension.
-----------------	--

Returns

True if file exists, False otherwise.

check_sources()

```
check_sources (
    filepaths,
    path = None )
```

Checks each of the sources provided and removes any bad sources.

Any filepaths or source files that are not images or do not exist get removed.

Parameters

<i>filepaths</i>	List of file paths.
<i>path</i>	A path to the images, if it is provided.

Returns

True if all/some sources are good, False if all sources are bad.

extract_color_palettes()

```
extract_color_palettes ( )
```

Handles color extraction from image(s).

handle_args()

```
handle_args ( )
```

Handles the arguments passed to PyPalEx.

main()

```
main ( )
```

Main script function.

set_global_args()

```
set_global_args (
    args )
```

Sets the global variables using the arguments.

Parameters

<i>args</i>	User-supplied arguments.
-------------	--------------------------

setup_argument_parser()

```
setup_argument_parser ( )
```

Sets up the argument parser for command line arguments.

Returns

A command line argument parsing object.

5.2.2 Variable Documentation**EXTRACTORS**

```
list EXTRACTORS = [ ]
```

List of Extractor class objects for each individual image.

FILENAMES

```
list FILENAMES = [ ]
```

List of image filenames.

OUTPUT_FILEPATHS

```
list OUTPUT_FILEPATHS = [ ]
```

List of output file path(s) for each image.

OUTPUT_PATH

```
str OUTPUT_PATH = ''
```

The path to the output directory where all JSON files will be saved.

OUTPUT_TAIL

```
str OUTPUT_TAIL = "-color_palette.json"
```

The tail to append to each output filepath.

PASTEL_D

```
bool PASTEL_D = False
```

Flag to convert dark color palette to pastel.

PASTEL_L

```
bool PASTEL_L = False
```

Flag to convert light color palette to pastel.

PASTEL_N

```
bool PASTEL_N = False
```

Flag to convert normal color palette to pastel.

PROPER_IMAGES

```
list PROPER_IMAGES = []
```

List of real/existing image file path(s).

SAVE_CHECK

```
bool SAVE_CHECK = False
```

Flag to check if user wants to save extracted color palettes.

SHOW_PREVIEW

```
bool SHOW_PREVIEW = False
```

Flag to show a preview of extracted palettes.

5.3 pypalex.arg_messages Namespace Reference**Functions**

- [bad_source_message](#) ()
Generates an error message if the sources provided were not images.
- [bad_path_message](#) ()
Generates an error message if the directory provided is not a valid directory.
- [no_args_help_message](#) ()
Generates a help message if no arguments were presented.

5.3.1 Function Documentation

bad_path_message()

`bad_path_message ()`

Generates an error message if the directory provided is not a valid directory.

Returns

The "bad directory" message.

bad_source_message()

`bad_source_message ()`

Generates an error message if the sources provided were not images.

Returns

The "bad sources" message.

no_args_help_message()

`no_args_help_message ()`

Generates a help message if no arguments were presented.

Returns

The "no arguments" help message.

5.4 pypalex.constants Namespace Reference

Variables

- list `BLACK_RGB` = [0, 0, 0]
- list `WHITE_RGB` = [255, 255, 255]
- list `RED_RGB` = [255, 0, 0]
- list `YELLOW_RGB` = [255, 234, 0]
- list `GREEN_RGB` = [0, 255, 0]
- list `CYAN_RGB` = [0, 255, 255]
- list `BLUE_RGB` = [0, 0, 255]
- list `MAGENTA_RGB` = [255, 0, 255]
- int `BLACK_HEX` = 0x000000
- int `WHITE_HEX` = 0xFFFFFF
- int `RED_HEX` = 0xFF0000
- int `YELLOW_HEX` = 0xFFEA00
- int `GREEN_HEX` = 0x00FF00

- int [CYAN_HEX](#) = 0x00FFFF
- int [BLUE_HEX](#) = 0x0000FF
- int [MAGENTA_HEX](#) = 0xFF00FF
- int [RED_HUE](#) = 0
- int [YELLOW_HUE](#) = 55
- int [GREEN_HUE](#) = 120
- int [CYAN_HUE](#) = 180
- int [BLUE_HUE](#) = 240
- int [MAGENTA_HUE](#) = 300
- list [RED_HUE_RANGE_MAX](#) = [330, 360]
- list [RED_HUE_RANGE_MIN](#) = [0, 25]
- list [YELLOW_HUE_RANGE](#) = [25, 64]
- list [GREEN_HUE_RANGE](#) = [64, 170]
- list [CYAN_HUE_RANGE](#) = [170, 210]
- list [BLUE_HUE_RANGE](#) = [210, 260]
- list [MAGENTA_HUE_RANGE](#) = [260, 330]
- list [BLACK_BRIGHTNESS_RANGE](#) = [0.0, 35.0]
- list [DARK_BRIGHTNESS_RANGE](#) = [35.0, 55.0]
- list [NORM_BRIGHTNESS_RANGE](#) = [55.0, 80.0]
- list [LIGHT_BRIGHTNESS_RANGE](#) = [80.0, 100.0]
- list [SATURATION_TOLERANCE_RANGE](#) = [10.0, 15.0]
- list [PASTEL_SATURATION_RANGE](#) = [15.0, 75.0]
- list [PASTEL_BRIGHTNESS_RANGE](#) = [65.0, 95.0]

5.4.1 Variable Documentation

BLACK_BRIGHTNESS_RANGE

```
list BLACK_BRIGHTNESS_RANGE = [0.0, 35.0]
```

BLACK_HEX

```
int BLACK_HEX = 0x000000
```

BLACK_RGB

```
list BLACK_RGB = [0, 0, 0]
```

BLUE_HEX

```
int BLUE_HEX = 0x0000FF
```

BLUE_HUE

```
int BLUE_HUE = 240
```

BLUE_HUE_RANGE

```
list BLUE_HUE_RANGE = [210, 260]
```

BLUE_RGB

```
list BLUE_RGB = [0, 0, 255]
```

CYAN_HEX

```
int CYAN_HEX = 0x00FFFF
```

CYAN_HUE

```
int CYAN_HUE = 180
```

CYAN_HUE_RANGE

```
list CYAN_HUE_RANGE = [170, 210]
```

CYAN_RGB

```
list CYAN_RGB = [0, 255, 255]
```

DARK_BRIGHTNESS_RANGE

```
list DARK_BRIGHTNESS_RANGE = [35.0, 55.0]
```

GREEN_HEX

```
int GREEN_HEX = 0x00FF00
```

GREEN_HUE

```
int GREEN_HUE = 120
```

GREEN_HUE_RANGE

```
list GREEN_HUE_RANGE = [64, 170]
```

GREEN_RGB

```
list GREEN_RGB = [0, 255, 0]
```

LIGHT_BRIGHTNESS_RANGE

```
list LIGHT_BRIGHTNESS_RANGE = [80.0, 100.0]
```

MAGENTA_HEX

```
int MAGENTA_HEX = 0xFF00FF
```

MAGENTA_HUE

```
int MAGENTA_HUE = 300
```

MAGENTA_HUE_RANGE

```
list MAGENTA_HUE_RANGE = [260, 330]
```

MAGENTA_RGB

```
list MAGENTA_RGB = [255, 0, 255]
```

NORM_BRIGHTNESS_RANGE

```
list NORM_BRIGHTNESS_RANGE = [55.0, 80.0]
```

PASTEL_BRIGHTNESS_RANGE

```
list PASTEL_BRIGHTNESS_RANGE = [65.0, 95.0]
```

PASTEL_SATURATION_RANGE

```
list PASTEL_SATURATION_RANGE = [15.0, 75.0]
```

RED_HEX

```
int RED_HEX = 0xFF0000
```

RED_HUE

```
int RED_HUE = 0
```

RED_HUE_RANGE_MAX

```
list RED_HUE_RANGE_MAX = [330, 360]
```

RED_HUE_RANGE_MIN

```
list RED_HUE_RANGE_MIN = [0, 25]
```

RED_RGB

```
list RED_RGB = [255, 0, 0]
```

SATURATION_TOLERANCE_RANGE

```
list SATURATION_TOLERANCE_RANGE = [10.0, 15.0]
```

WHITE_HEX

```
int WHITE_HEX = 0xFFFFFF
```

WHITE_RGB

```
list WHITE_RGB = [255, 255, 255]
```

YELLOW_HEX

```
int YELLOW_HEX = 0xFFEA00
```

YELLOW_HUE

```
int YELLOW_HUE = 55
```

YELLOW_HUE_RANGE

```
list YELLOW_HUE_RANGE = [25, 64]
```

YELLOW_RGB

```
list YELLOW_RGB = [255, 234, 0]
```

5.5 pypalex.conversion_utils Namespace Reference**Functions**

- [rgb_to_hsv](#) (rgb_array)
Converts RGB array [r,g,b] to HSV array [h,s,v].
- [hsv_to_hex](#) (hsv_array)
Convert HSV array [h,s,v] to HEX string 'ffffff'.
- [hex_to_rgb](#) (hex_str)
Convert HEX string 'ffffff' to RGB array [r,g,b].
- [hsv_to_rgb](#) (hsv_array)
Convert HSV array [h,s,v] to RGB array [r,g,b].
- [rgb_to_hex](#) (rgb_array)
Convert RGB array [r,g,b] to HEX string 'ffffff'.

5.5.1 Function Documentation**hex_to_rgb()**

```
hex_to_rgb (
    hex_str )
```

Convert HEX string 'ffffff' to RGB array [r,g,b].

HEX string is in the set ["000000", "ffffff"]. RGB where [r,g,b] are in the set [0, 255].

Parameters

<i>hex_str</i>	HEX string 'ffffff'.
----------------	----------------------

Returns

RGB array [r,g,b].

hsv_to_hex()

```
hsv_to_hex (
    hsv_array )
```

Convert HSV array [h,s,v] to HEX string 'ffffff'.

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. HEX string is in the set ["000000", "ffffff"].

Parameters

<i>hsv_array</i>	HSV array [h,s,v].
------------------	--------------------

Returns

A HEX string.

hsv_to_rgb()

```
hsv_to_rgb (
    hsv_array )
```

Convert HSV array [h,s,v] to RGB array [r,g,b].

HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0]. RGB where [r,g,b] are in the set [0, 255].
Formula adapted from <https://www.rapidtables.com/convert/color/hsv-to-rgb.html>

Parameters

<i>hsv_array</i>	HSV array [h,s,v].
------------------	--------------------

Returns

RGB array [r,g,b].

rgb_to_hex()

```
rgb_to_hex (
    rgb_array )
```

Convert RGB array [r,g,b] to HEX string 'ffffff'.

RGB where [r,g,b] are in the set [0, 255]. HEX string is in the set ["000000", "ffffff"].

Parameters

<i>rgb_array</i>	RGB array [r,g,b].
------------------	--------------------

Returns

A HEX string.

rgb_to_hsv()

```
rgb_to_hsv (
    rgb_array )
```


Converts RGB array [r,g,b] to HSV array [h,s,v].

RGB where [r,g,b] are in the set [0, 255]. HSV where h is in the set [0, 359] and s, v are in the set [0.0, 100.0].
Formula adapted from <https://www.rapidtables.com/convert/color/rgb-to-hsv.html>

Parameters

<i>rgb_array</i>	RGB array [r,g,b].
------------------	--------------------

Returns

HSV array [h,s,v].

5.6 pypalex.extraction_utils Namespace Reference

Functions

- [extract_ratios](#) (hsv_img_matrix_2d)
Extracts the ratios of hues per pixel.
- [construct_base_color_dictionary](#) (hsv_img_matrix_2d)
Constructs dictionary of base colors from an array of HSV pixel values.
- [extract_color_palettes](#) (base_color_dict)
Extracts dominant light, normal, dark color palettes from each of the base colors.
- [check_missing_colors](#) (base_color_dict, extracted_colors_dict)
Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.
- [generate_remaining_colors](#) (extracted_colors_dict, ratios)
Generate the remaining black and white, and background and foreground colors.
- [extract_color_types](#) (hsv_base_color_matrix)
Extracts the dominant color types from a base color.
- [get_left_and_right_colors](#) (origin_color_name)
Gets the color names of the colors that are to the left and right of the originating color.
- [borrow_color](#) (extracted_colors_dict, origin, borrow_left, borrow_right)
Borrows a color from one of the extracted color types of the base colors.
- [get_dominant_hue](#) (extracted_colors_dict, ratios)
Calculates the dominant hue.
- [generate_black_and_white](#) (dominant_hue)
Generates black and white color types using the dominant hue.
- [generate_background_and_foreground](#) (dominant_hue, complementary_hue)
Generates the background and foreground colors.
- [sort_by_sat_and_bright_value](#) (hsv_base_color_matrix)
Sorts the colors by their saturation and brightness values.
- [extract_dominant_color](#) (hsv_color_type_matrix)
Extracts the dominant color from a color type.
- [check_missing_color_types](#) (light_color, norm_color, dark_color, black_color, achromatic_light, achromatic_norm, achromatic_dark, achromatic_black)
Checks to make sure all the color types have been properly set.
- [calculate_centroid](#) (hsv_color_type_matrix)
Calculates the centroid for a color type.
- [find_closest_to_centroid](#) (hsv_color_type_matrix, centroid)
Finds a color from a color type that is closest to the centroid.

5.6.1 Function Documentation

borrow_color()

```
borrow_color (
    extracted_colors_dict,
    origin,
    borrow_left,
    borrow_right )
```

Borrows a color from one of the extracted color types of the base colors.

Parameters

<i>extracted_colors_dict</i>	A Dictionary of extracted colors.
<i>origin</i>	The name of the originating color.
<i>borrow_left</i>	The name of the color to borrow from, to the left of origin.
<i>borrow_right</i>	The name of the color to borrow from, to the right of origin.

Returns

A numpy array of a borrowed color.

calculate_centroid()

```
calculate_centroid (
    hsv_color_type_matrix )
```

Calculates the centroid for a color type.

The centroid is basically the average color of a set of colors in [h,s,v] format. The centroid is a point in 3-dimensional space. The following sources were used to make this algorithm: <http://mkweb.bcgsc.ca/color-summarizer/?faq#averagehue> and <https://stackoverflow.com/a/8170595/17047816>

Parameters

<i>hsv_color_type_matrix</i>	A 2D numpy array of a color type in [h,s,v] format.
------------------------------	---

Returns

List of centroid color values in [h,s,l] format.

check_missing_color_types()

```
check_missing_color_types (
    light_color,
    norm_color,
    dark_color,
```

```

    black_color,
    achromatic_light,
    achromatic_norm,
    achromatic_dark,
    achromatic_black )

```

Checks to make sure all the color types have been properly set.

If a color type is missing, then it will be derived from the existing color types.

Note

I'm using the normalization formula from <https://stats.stackexchange.com/a/281164>

Parameters

<i>light_color</i>	A numpy array of a light color type in [h,s,v] format.
<i>norm_color</i>	A numpy array of a normal color type in [h,s,v] format.
<i>dark_color</i>	A numpy array of a dark color type in [h,s,v] format.
<i>black_color</i>	A numpy array of a black color type in [h,s,v] format.
<i>achromatic_light</i>	A numpy array of an achromatic light color type in [h,s,v] format.
<i>achromatic_norm</i>	A numpy array of an achromatic normal color type in [h,s,v] format.
<i>achromatic_dark</i>	A numpy array of an achromatic dark color type in [h,s,v] format.
<i>achromatic_black</i>	A numpy array of an achromatic black color type in [h,s,v] format.

`check_missing_colors()`

```

check_missing_colors (
    base_color_dict,
    extracted_colors_dict )

```

Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.

Parameters

<i>base_color_dict</i>	A dictionary of 2D numpy arrays for each of the base colors.
<i>extracted_colors_dict</i>	A Dictionary of extracted colors.

`construct_base_color_dictionary()`

```

construct_base_color_dictionary (
    hsv_img_matrix_2d )

```

Constructs dictionary of base colors from an array of HSV pixel values.

Base colors are classified as [red, yellow, green, cyan, blue, magenta].

Parameters

<i>hsv_img_matrix_2d</i>	A 2D numpy array of pixels from an image, in [h,s,v] format.
--------------------------	--

Returns

Dictionary of base colors.

extract_color_palettes()

```
extract_color_palettes (  
    base_color_dict )
```

Extracts dominant light, normal, dark color palettes from each of the base colors.

Parameters

<i>base_color_dict</i>	A dictionary of 2D numpy arrays for each of the base colors.
------------------------	--

Returns

Dictionary of light, normal, dark color palettes for each of the base colors.

extract_color_types()

```
extract_color_types (  
    hsv_base_color_matrix )
```

Extracts the dominant color types from a base color.

A color type is either a light, normal, or dark version of a base color.

Parameters

<i>hsv_base_color_matrix</i>	A 2D numpy array of a base color where every element is a list in [h,s,v] format.
------------------------------	---

Returns

List of dominant color types, where each color type is a numpy array in [h,s,v] format.

extract_dominant_color()

```
extract_dominant_color (  
    hsv_color_type_matrix )
```

Extracts the dominant color from a color type.

A color type is either a light, normal, or dark version of a base color.

Parameters

<i>hsv_color_type_matrix</i>	A 2D numpy array of a color type where every element is a list in [h,s,v] format.
------------------------------	---

Returns

A numpy array of a dominant color from a color type in [h,s,v] format.

extract_ratios()

```
extract_ratios (
    hsv_img_matrix_2d )
```

Extracts the ratios of hues per pixel.

Parameters

<i>hsv_img_matrix_2d</i>	A 2D numpy array of pixels from an image in [h,s,v] format.
--------------------------	---

Returns

Dictionary of hue ratios (percentage) in set [0.0, 100.0]

find_closest_to_centroid()

```
find_closest_to_centroid (
    hsv_color_type_matrix,
    centroid )
```

Finds a color from a color type that is closest to the centroid.

The distance between the centroid color and each of the other individual colors is calculated in 3-dimensional space using the Euclidean Distance formula from the following sources: <https://stackoverflow.com/a/35114586/17047816> and <https://byjus.com/maths/distance-between-two-points-3d/>.

Parameters

<i>hsv_color_type_matrix</i>	A 2D numpy array of a color type where every element is a list in [h,s,v] format.
<i>centroid</i>	List of centroid color values in [h,s,l] format.

Returns

List of all the colors in [h,s,v] format that are the shortest distance away from the centroid.

generate_background_and_foreground()

```
generate_background_and_foreground (
```

```
dominant_hue,  
complementary_hue )
```

Generates the background and foreground colors.

The background and foreground colors are based on the dominant hue in an image and it's complimentary hue. The saturation and brightness values for the background and foreground colors need to be hardcoded to be easier to look at.

Parameters

<i>dominant_hue</i>	The dominant hue of an image.
<i>complementary_hue</i>	The complimentary hue to the dominant hue.

Returns

Numpy array of light and dark background and foreground colors in [h,s,v] format.

generate_black_and_white()

```
generate_black_and_white (   
    dominant_hue )
```

Generates black and white color types using the dominant hue.

The saturation and brightness values, for the black and white color types, needs to be hardcoded in order to not interfere with the background and foreground colors.

Parameters

<i>dominant_hue</i>	The dominant hue of an image.
---------------------	-------------------------------

Returns

List of black and white color types in [h,s,v] format.

generate_remaining_colors()

```
generate_remaining_colors (   
    extracted_colors_dict,   
    ratios )
```

Generate the remaining black and white, and background and foreground colors.

Parameters

<i>extracted_colors_dict</i>	A Dictionary of extracted colors.
<i>ratios</i>	A Dictionary of ratios of the base colors in the image.

get_dominant_hue()

```
get_dominant_hue (
    extracted_colors_dict,
    ratios )
```

Calculates the dominant hue.

The dominant hue, also referred to as the average hue, is based on the color ratios and the colors extracted from an image.

Parameters

<i>extracted_colors_dict</i>	A Dictionary of extracted colors.
<i>ratios</i>	A Dictionary of ratios of the base colors in the image.

Returns

The dominant hue in an image.

get_left_and_right_colors()

```
get_left_and_right_colors (
    origin_color_name )
```

Gets the color names of the colors that are to the left and right of the originating color.

There are two ways to think about left and right on a color wheel: from the inside looking outward and from the outside looking inward. This has an effect on how we think of the linear format of the color wheel. For this package we will think about left and right colors using the latter option.

Parameters

<i>origin_color_name</i>	The name of the originating color.
--------------------------	------------------------------------

Returns

List of color names that are to the left and right of the originating color.

sort_by_sat_and_bright_value()

```
sort_by_sat_and_bright_value (
    hsv_base_color_matrix )
```

Sorts the colors by their saturation and brightness values.

A color type is either a light, normal, dark, black or achromatic version of a base color.

Parameters

<i>hsv_base_color_matrix</i>	A 2D numpy array of a base color, where each element is a list in [h,s,v] format.
------------------------------	---

Returns

A list of color types, where each element is a 2D numpy array of a color type whose elements are a list in [h,s,v] format.

5.7 pypalex.Extractor Namespace Reference

Classes

- class [Extractor](#)
Extracts colors given a matrix of HSV values extracted from an image.

5.8 pypalex.file_utils Namespace Reference

Functions

- [save_palette_to_file](#) (color_palette, output_filepath)
Saves color palette to json file.
- [save_default_scheme_to_file](#) (color_palette, output_filepath)
Saves color palette to json file as default color schemes.

5.8.1 Function Documentation

save_default_scheme_to_file()

```
save_default_scheme_to_file (
    color_palette,
    output_filepath )
```

Saves color palette to json file as default color schemes.

Constructs 2 default color schemes, light and dark, using the color palettes and saves them to a json file.

Note

If a file with the same name already exists, it is overwritten.

Parameters

<i>color_palette</i>	Dictionary of light, normal, and dark color palettes.
<i>output_filepath</i>	Output file path with filename of where to store color palette.

save_palette_to_file()

```
save_palette_to_file (
    color_palette,
    output_filepath )
```

Saves color palette to json file.

Note

If a file with the same name already exists, it is overwritten.

Parameters

<i>color_palette</i>	Dictionary of light, normal, and dark color palettes.
<i>output_filepath</i>	Output file path with filename of where to store color palette.

5.9 pypalex.image_utils Namespace Reference

Functions

- [process_image](#) (image)
Processes PIL Image object.
- [rescale_image](#) (image)
Rescales image to a smaller sampling size while maintaining aspect ration.
- [process_helper](#) (rgb_matrix_2d)
Helper function for multiprocessing conversion operations.

5.9.1 Function Documentation

process_helper()

```
process_helper (
    rgb_matrix_2d )
```

Helper function for multiprocessing conversion operations.

Helps convert from [r,g,b] to [h,s,v].

Parameters

<i>rgb_matrix_2d</i>	A 2D matrix of rgb values.
----------------------	----------------------------

Returns

A numpy array/2D matrix of converted [h,s,v] values.

process_image()

```
process_image (
    image )
```

Processes PIL Image object.

Multiprocessing example from: <https://stackoverflow.com/a/45555516>

Parameters

<i>image</i>	PIL Image object.
--------------	-------------------

Returns

2D numpy array of [h,s,v] arrays (pixels) from image.

rescale_image()

```
rescale_image (
    image )
```

Rescales image to a smaller sampling size while maintaining aspect ration.

Note

The math behind rescaling the image came from: <https://math.stackexchange.com/a/3078131>

Parameters

<i>image</i>	PIL Image object.
--------------	-------------------

Returns

Tuple of the new width and height of image.

5.10 pypalex.print_utils Namespace Reference

Functions

- [print_default_scheme_preview](#) (hex_color_palette)
Prints the default color schemes to the terminal.
- [get_color_escape](#) (rgb_array, background=False)
Constructs ANSI color escape code based on an RGB list.
- [get_rgb_palette](#) (hex_color_palette)
Constructs an RGB [r,g,b] palette dictionary using a hex palette dictionary.
- [get_ansi_color_codes](#) (rgb_color_palette)
Constructs a ANSI escape code dictionary using a RGB [r,g,b] palette dictionary.
- [generate_panes](#) (background_ansi_color, ansi_colors1, ansi_colors2)
Generates panes based on two sets of ANSI color escape codes.

5.10.1 Function Documentation

generate_panes()

```
generate_panes (
    background_ansi_color,
    ansi_colors1,
    ansi_colors2 )
```

Generates panes based on two sets of ANSI color escape codes.

Note

The terminal needs to be able to display ASCII characters and ANSI colors for this to be useful.

Parameters

<i>background_ansi_color</i>	The background ANSI color escape code.
<i>ansi_colors1</i>	List of ANSI color escape codes.
<i>ansi_colors2</i>	List of ANSI color escape codes.

Returns

List of strings of panes with ASCII and ANSI escape codes.

get_ansi_color_codes()

```
get_ansi_color_codes (
    rgb_color_palette )
```

Constructs a ANSI escape code dictionary using a RGB [r,g,b] palette dictionary.

Parameters

<i>rgb_color_palette</i>	A dictionary of light, normal and dark color palettes in RGB [r,g,b] format.
--------------------------	--

Returns

A dictionary of ANSI color escape codes.

get_color_escape()

```
get_color_escape (
    rgb_array,
    background = False )
```

Constructs ANSI color escape code based on an RGB list.

An RGB [r,g,b] list is used to generate an ANSI escape code of the RGB color for use in the terminal CLI. The basic format for these codes depends on if it will be used for foreground or background color. Use \033[38;2;r;g;bm for the foreground color. Use \033[48;2;r;g;bm for the background color.

Note

For more information about these ANSI escape codes, here are some sources: [https://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences/33206814#](https://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences/33206814#https://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences/33206814)
<https://stackoverflow.com/questions/45782766/color-python-output-given-rrggbg-hex-v>

Parameters

<i>rgb_array</i>	RGB array [r,g,b].
<i>background</i>	Flag for if the RGB color is for a background or not.

Returns

ANSI escape code of the RGB color.

get_rgb_palette()

```
get_rgb_palette (
    hex_color_palette )
```

Constructs an RGB [r,g,b] palette dictionary using a hex palette dictionary.

Parameters

<i>hex_color_palette</i>	A dictionary of color palettes in hex format.
--------------------------	---

Returns

A dictionary of colors in RGB [r,g,b] format.

print_default_scheme_preview()

```
print_default_scheme_preview (
    hex_color_palette )
```

Prints the default color schemes to the terminal.

Prints a preview of the extracted color palettes to the user's terminal screen using ANSI escape codes.

Note

The terminal needs to be able to display ASCII characters and ANSI colors for this to work.

Parameters

<i>hex_color_palette</i>	A dictionary of light, normal, and dark color palettes in hex format.
--------------------------	---

6 Class Documentation

6.1 Extractor Class Reference

Extracts colors given a matrix of HSV values extracted from an image.

Public Member Functions

- [__init__](#) (self, [hsv_img_matrix_2d](#), [output_filepath](#), [pastel_light](#)=False, [pastel_normal](#)=False, [pastel_dark](#)=False)
Extractor Constructor.
- [run](#) (self)
Main method for Extractor class.
- [check_pastel_conversion](#) (self)
Checks to see if any of the palettes should be converted to pastel.
- [construct_palette_dictionary](#) (self)
Constructs a dictionary of all the extracted color palettes in hex format.
- [construct_scheme_dictionary](#) (self)
Constructs a dictionary of color schemes by combining color palettes.
- [convert_pastel_light](#) (self)
Converts light palette to pastel.
- [convert_pastel_normal](#) (self)
Converts normal palette to pastel.
- [convert_pastel_dark](#) (self)
Converts dark palette to pastel.
- [convert_pastel](#) (self, [hsv_color](#))
Converts/normalizes HSV color to pastel.

Public Attributes

- [hsv_img_matrix_2d](#)
A 2D numpy array of pixels from an image in [h,s,v] format.
- [output_filepath](#)
Output file path with filename of where to store color palette.
- [pastel_light](#)
Flag to convert light color palette to pastel.
- [pastel_normal](#)
Flag to convert normal color palette to pastel.
- [pastel_dark](#)
Flag to convert dark color palette to pastel.
- [ratio_dict](#)
A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.
- [base_color_dict](#)
A dictionary of 2D numpy arrays for each of the 6 base colors.
- [extracted_colors_dict](#)
A dictionary of extracted colors in [h,s,v] format.
- [palette_dict](#)
A dictionary of light, normal, and dark color palettes in hex format.

6.1.1 Detailed Description

Extracts colors given a matrix of HSV values extracted from an image.

6.1.2 Constructor & Destructor Documentation

`__init__()`

```
__init__ (
    self,
    hsv_img_matrix_2d,
    output_filepath,
    pastel_light = False,
    pastel_normal = False,
    pastel_dark = False )
```

Extractor Constructor.

Parameters

<i>self</i>	The object pointer.
<i>hsv_img_matrix_2d</i>	A 2D numpy array of pixels from an image in [h,s,v] format.
<i>output_filepath</i>	Output file path with filename of where to store color palette.
<i>pastel_light</i>	Flag to convert light color palette to pastel.
<i>pastel_normal</i>	Flag to convert normal color palette to pastel.
<i>pastel_dark</i>	Flag to convert dark color palette to pastel.

6.1.3 Member Function Documentation

`check_pastel_conversion()`

```
check_pastel_conversion (
    self )
```

Checks to see if any of the palettes should be converted to pastel.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

`construct_palette_dictionary()`

```
construct_palette_dictionary (
    self )
```

Constructs a dictionary of all the extracted color palettes in hex format.

The extracted color palettes are organized in the dictionary as follows: light background, light foreground, dark background, dark foreground, light palette, normal palette, dark palette.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

construct_scheme_dictionary()

```
construct_scheme_dictionary (  
    self )
```

Constructs a dictionary of color schemes by combining color palettes.

Light color scheme contains the normal and dark color palettes. Dark color scheme contains the normal and light color palettes.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

Returns

A dictionary of light and dark color schemes.

convert_pastel()

```
convert_pastel (  
    self,  
    hsv_color )
```

Converts/normalizes HSV color to pastel.

For values x in range $[a, b]$, values x can be normalized to the new range $[y, z]$ with the following equation: $\text{new_}x = y + ((x-a) / (b-a)) * (z-y)$

Note

I'm using the normalization formula from <https://stats.stackexchange.com/a/281164>

Parameters

<i>self</i>	The object pointer.
<i>hsv_color</i>	List HSV color to be converted to pastel.

convert_pastel_dark()

```
convert_pastel_dark (  
    self )
```

Converts dark palette to pastel.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

convert_pastel_light()

```
convert_pastel_light (  
    self )
```

Converts light palette to pastel.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

convert_pastel_normal()

```
convert_pastel_normal (  
    self )
```

Converts normal palette to pastel.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

run()

```
run (  
    self )
```

Main method for Extractor class.

Performs extraction of colors.

Parameters

<i>self</i>	The object pointer.
-------------	---------------------

6.1.4 Member Data Documentation**base_color_dict**

```
base_color_dict
```

A dictionary of 2D numpy arrays for each of the 6 base colors.

extracted_colors_dict

```
extracted_colors_dict
```

A dictionary of extracted colors in [h,s,v] format.

hsv_img_matrix_2d

```
hsv_img_matrix_2d
```

A 2D numpy array of pixels from an image in [h,s,v] format.

output_filepath

```
output_filepath
```

Output file path with filename of where to store color palette.

palette_dict

```
palette_dict
```

A dictionary of light, normal, and dark color palettes in hex format.

pastel_dark

```
pastel_dark
```

Flag to convert dark color palette to pastel.

pastel_light

```
pastel_light
```

Flag to convert light color palette to pastel.

pastel_normal

```
pastel_normal
```

Flag to convert normal color palette to pastel.

ratio_dict

ratio_dict

A dictionary that holds the ratio of base colors in an image and is used to identify the dominant color in an image.

The documentation for this class was generated from the following file:

- [Extractor.py](#)

7 File Documentation

7.1 __main__.py File Reference

Main script for PyPalEx.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.__main__](#)

Functions

- [main](#) ()
Main script function.
- [handle_args](#) ()
Handles the arguments passed to PyPalEx.
- [extract_color_palettes](#) ()
Handles color extraction from image(s).
- [setup_argument_parser](#) ()
Sets up the argument parser for command line arguments.
- [check_sources](#) (filepaths, path=None)
Checks each of the sources provided and removes any bad sources.
- [check_path](#) (path)
Check the path to make sure it exists.
- [set_global_args](#) (args)
Sets the global variables using the arguments.
- [check_source](#) (filepath)
Checks to make sure the path leads to a file.

Variables

- list `EXTRACTORS` = []
List of Extractor class objects for each individual image.
- list `PROPER_IMAGES` = []
List of real/existing image file path(s).
- list `FILENAMES` = []
List of image filenames.
- list `OUTPUT_FILEPATHS` = []
List of output file path(s) for each image.
- str `OUTPUT_PATH` = "
The path to the output directory where all JSON files will be saved.
- str `OUTPUT_TAIL` = "-color_palette.json"
The tail to append to each output filepath.
- bool `SAVE_CHECK` = False
Flag to check if user wants to save extracted color palettes.
- bool `SHOW_PREVIEW` = False
Flag to show a preview of extracted palettes.
- bool `PASTEL_L` = False
Flag to convert light color palette to pastel.
- bool `PASTEL_N` = False
Flag to convert normal color palette to pastel.
- bool `PASTEL_D` = False
Flag to convert dark color palette to pastel.

7.1.1 Detailed Description

Main script for PyPalEx.

Used to run from the Command Line.

7.1.2 Author(s)

- Created by AI Timofeyev on February 2, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on March 22, 2023.
- Modified by AI Timofeyev on March 26, 2023.
- Modified by AI Timofeyev on April 7, 2023.
- Modified by AI Timofeyev on June 10, 2024.

7.2 `arg_messages.py` File Reference

Archive of messages to display for arguments supplied by user.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.arg_messages](#)

Functions

- [bad_source_message](#) ()
Generates an error message if the sources provided were not images.
- [bad_path_message](#) ()
Generates an error message if the directory provided is not a valid directory.
- [no_args_help_message](#) ()
Generates a help message if no arguments were presented.

7.2.1 Detailed Description

Archive of messages to display for arguments supplied by user.

7.2.2 Author(s)

- Created by Al Timofeyev on March 3, 2022.
- Modified by Al Timofeyev on April 21, 2022.
- Modified by Al Timofeyev on March 6, 2023.

7.3 constants.py File Reference

A collection of constants for PyPalEx.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.constants](#)

Variables

- list `BLACK_RGB` = [0, 0, 0]
- list `WHITE_RGB` = [255, 255, 255]
- list `RED_RGB` = [255, 0, 0]
- list `YELLOW_RGB` = [255, 234, 0]
- list `GREEN_RGB` = [0, 255, 0]
- list `CYAN_RGB` = [0, 255, 255]
- list `BLUE_RGB` = [0, 0, 255]
- list `MAGENTA_RGB` = [255, 0, 255]
- int `BLACK_HEX` = 0x000000
- int `WHITE_HEX` = 0xFFFFFFFF
- int `RED_HEX` = 0xFF0000
- int `YELLOW_HEX` = 0xFFEA00
- int `GREEN_HEX` = 0x00FF00
- int `CYAN_HEX` = 0x00FFFF
- int `BLUE_HEX` = 0x0000FF
- int `MAGENTA_HEX` = 0xFF00FF
- int `RED_HUE` = 0
- int `YELLOW_HUE` = 55
- int `GREEN_HUE` = 120
- int `CYAN_HUE` = 180
- int `BLUE_HUE` = 240
- int `MAGENTA_HUE` = 300
- list `RED_HUE_RANGE_MAX` = [330, 360]
- list `RED_HUE_RANGE_MIN` = [0, 25]
- list `YELLOW_HUE_RANGE` = [25, 64]
- list `GREEN_HUE_RANGE` = [64, 170]
- list `CYAN_HUE_RANGE` = [170, 210]
- list `BLUE_HUE_RANGE` = [210, 260]
- list `MAGENTA_HUE_RANGE` = [260, 330]
- list `BLACK_BRIGHTNESS_RANGE` = [0.0, 35.0]
- list `DARK_BRIGHTNESS_RANGE` = [35.0, 55.0]
- list `NORM_BRIGHTNESS_RANGE` = [55.0, 80.0]
- list `LIGHT_BRIGHTNESS_RANGE` = [80.0, 100.0]
- list `SATURATION_TOLERANCE_RANGE` = [10.0, 15.0]
- list `PASTEL_SATURATION_RANGE` = [15.0, 75.0]
- list `PASTEL_BRIGHTNESS_RANGE` = [65.0, 95.0]

7.3.1 Detailed Description

A collection of constants for PyPalEx.

7.3.2 Author(s)

- Created by AI Timofeyev on February 2, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on May 31, 2024.
- Modified by AI Timofeyev on June 10, 2024.

7.4 conversion_utils.py File Reference

Utilities for converting between RGB, HSV, HEX.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.conversion_utils](#)

Functions

- [rgb_to_hsv](#) (rgb_array)
Converts RGB array [r,g,b] to HSV array [h,s,v].
- [hsv_to_hex](#) (hsv_array)
Convert HSV array [h,s,v] to HEX string 'ffffff'.
- [hex_to_rgb](#) (hex_str)
Convert HEX string 'ffffff' to RGB array [r,g,b].
- [hsv_to_rgb](#) (hsv_array)
Convert HSV array [h,s,v] to RGB array [r,g,b].
- [rgb_to_hex](#) (rgb_array)
Convert RGB array [r,g,b] to HEX string 'ffffff'.

7.4.1 Detailed Description

Utilities for converting between RGB, HSV, HEX.

7.4.2 Author(s)

- Created by AI Timofeyev on February 2, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on April 5, 2023.

7.5 extraction_utils.py File Reference

Utilities for extracting colors from the image.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.extraction_utils](#)

Functions

- [extract_ratios](#) (hsv_img_matrix_2d)
Extracts the ratios of hues per pixel.
- [construct_base_color_dictionary](#) (hsv_img_matrix_2d)
Constructs dictionary of base colors from an array of HSV pixel values.
- [extract_color_palettes](#) (base_color_dict)
Extracts dominant light, normal, dark color palettes from each of the base colors.
- [check_missing_colors](#) (base_color_dict, extracted_colors_dict)
Checks for any missing colors in the base color dictionary and borrows them from the surrounding colors.
- [generate_remaining_colors](#) (extracted_colors_dict, ratios)
Generate the remaining black and white, and background and foreground colors.
- [extract_color_types](#) (hsv_base_color_matrix)
Extracts the dominant color types from a base color.
- [get_left_and_right_colors](#) (origin_color_name)
Gets the color names of the colors that are to the left and right of the originating color.
- [borrow_color](#) (extracted_colors_dict, origin, borrow_left, borrow_right)
Borrows a color from one of the extracted color types of the base colors.
- [get_dominant_hue](#) (extracted_colors_dict, ratios)
Calculates the dominant hue.
- [generate_black_and_white](#) (dominant_hue)
Generates black and white color types using the dominant hue.
- [generate_background_and_foreground](#) (dominant_hue, complementary_hue)
Generates the background and foreground colors.
- [sort_by_sat_and_bright_value](#) (hsv_base_color_matrix)
Sorts the colors by their saturation and brightness values.
- [extract_dominant_color](#) (hsv_color_type_matrix)
Extracts the dominant color from a color type.
- [check_missing_color_types](#) (light_color, norm_color, dark_color, black_color, achromatic_light, achromatic_←_norm, achromatic_dark, achromatic_black)
Checks to make sure all the color types have been properly set.
- [calculate_centroid](#) (hsv_color_type_matrix)
Calculates the centroid for a color type.
- [find_closest_to_centroid](#) (hsv_color_type_matrix, centroid)
Finds a color from a color type that is closest to the centroid.

7.5.1 Detailed Description

Utilities for extracting colors from the image.

7.5.2 Author(s)

- Created by AI Timofeyev on February 10, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on March 22, 2023.
- Modified by AI Timofeyev on April 6, 2023.
- Modified by AI Timofeyev on May 31, 2024.
- Modified by AI Timofeyev on June 10, 2024.

7.6 Extractor.py File Reference

Extraction utility class for extracting colors from the image.

Classes

- class [Extractor](#)
Extracts colors given a matrix of HSV values extracted from an image.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.Extractor](#)

7.6.1 Detailed Description

Extraction utility class for extracting colors from the image.

7.6.2 Author(s)

- Created by AI Timofeyev on February 10, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on March 22, 2023.
- Modified by AI Timofeyev on April 5, 2023.
- Modified by AI Timofeyev on June 10, 2024.

7.7 file_utils.py File Reference

Utilities for file handling.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.file_utils](#)

Functions

- [save_palette_to_file](#) (color_palette, output_filepath)
Saves color palette to json file.
- [save_default_scheme_to_file](#) (color_palette, output_filepath)
Saves color palette to json file as default color schemes.

7.7.1 Detailed Description

Utilities for file handling.

Note

Potential point for contributors to add different output saving options.

7.7.2 Author(s)

- Created by AI Timofeyev on April 5, 2023.

7.8 image_utils.py File Reference

Utilities for processing image and file handling.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.image_utils](#)

Functions

- [process_image](#) (image)
Processes PIL Image object.
- [rescale_image](#) (image)
Rescales image to a smaller sampling size while maintaining aspect ration.
- [process_helper](#) (rgb_matrix_2d)
Helper function for multiprocessing conversion operations.

7.8.1 Detailed Description

Utilities for processing image and file handling.

7.8.2 Author(s)

- Created by AI Timofeyev on February 27, 2022.
- Modified by AI Timofeyev on April 21, 2022.
- Modified by AI Timofeyev on March 6, 2023.
- Modified by AI Timofeyev on April 5, 2023.
- Modified by AI Timofeyev on May 16, 2024.

7.9 print_utils.py File Reference

Utilities for printing preview to the screen.

Namespaces

- namespace [pypalex](#)
Python Palette [Extractor](#): extracts color palettes from images.
- namespace [pypalex.print_utils](#)

Functions

- [print_default_scheme_preview](#) (hex_color_palette)
Prints the default color schemes to the terminal.
- [get_color_escape](#) (rgb_array, background=False)
Constructs ANSI color escape code based on an RGB list.
- [get_rgb_palette](#) (hex_color_palette)
Constructs an RGB [r,g,b] palette dictionary using a hex palette dictionary.
- [get_ansi_color_codes](#) (rgb_color_palette)
Constructs a ANSI escape code dictionary using a RGB [r,g,b] palette dictionary.
- [generate_panes](#) (background_ansi_color, ansi_colors1, ansi_colors2)
Generates panes based on two sets of ANSI color escape codes.

7.9.1 Detailed Description

Utilities for printing preview to the screen.

Note

Potential point for contributors to add different printing options, maybe even a printing option that displays in a GUI.

7.9.2 Author(s)

- Created by Al Timofeyev on April 5, 2023.

Index

- [__init__](#)
 - [Extractor, 28](#)
 - [__main__.py, 33](#)
- [arg_messages.py, 34](#)
- [bad_path_message](#)
 - [pypalex.arg_messages, 8](#)
- [bad_source_message](#)
 - [pypalex.arg_messages, 8](#)
- [base_color_dict](#)
 - [Extractor, 31](#)
- [BLACK_BRIGHTNESS_RANGE](#)
 - [pypalex.constants, 9](#)
- [BLACK_HEX](#)
 - [pypalex.constants, 9](#)
- [BLACK_RGB](#)
 - [pypalex.constants, 9](#)
- [BLUE_HEX](#)
 - [pypalex.constants, 9](#)
- [BLUE_HUE](#)
 - [pypalex.constants, 9](#)
- [BLUE_HUE_RANGE](#)
 - [pypalex.constants, 9](#)
- [BLUE_RGB](#)
 - [pypalex.constants, 10](#)
- [borrow_color](#)
 - [pypalex.extraction_utils, 16](#)
- [calculate_centroid](#)
 - [pypalex.extraction_utils, 16](#)
- [check_missing_color_types](#)
 - [pypalex.extraction_utils, 16](#)
- [check_missing_colors](#)
 - [pypalex.extraction_utils, 17](#)
- [check_pastel_conversion](#)
 - [Extractor, 28](#)
- [check_path](#)
 - [pypalex.__main__, 4](#)
- [check_source](#)
 - [pypalex.__main__, 4](#)
- [check_sources](#)
 - [pypalex.__main__, 5](#)
- [constants.py, 35](#)
- [construct_base_color_dictionary](#)
 - [pypalex.extraction_utils, 17](#)
- [construct_palette_dictionary](#)
 - [Extractor, 28](#)
- [construct_scheme_dictionary](#)
 - [Extractor, 29](#)
- [conversion_utils.py, 37](#)
- [convert_pastel](#)
 - [Extractor, 29](#)
- [convert_pastel_dark](#)
 - [Extractor, 29](#)
- [convert_pastel_light](#)
 - [Extractor, 29](#)
- [Extractor, 31](#)
- [convert_pastel_normal](#)
 - [Extractor, 31](#)
- [CYAN_HEX](#)
 - [pypalex.constants, 10](#)
- [CYAN_HUE](#)
 - [pypalex.constants, 10](#)
- [CYAN_HUE_RANGE](#)
 - [pypalex.constants, 10](#)
- [CYAN_RGB](#)
 - [pypalex.constants, 10](#)
- [DARK_BRIGHTNESS_RANGE](#)
 - [pypalex.constants, 10](#)
- [extract_color_palettes](#)
 - [pypalex.__main__, 5](#)
 - [pypalex.extraction_utils, 18](#)
- [extract_color_types](#)
 - [pypalex.extraction_utils, 18](#)
- [extract_dominant_color](#)
 - [pypalex.extraction_utils, 18](#)
- [extract_ratios](#)
 - [pypalex.extraction_utils, 19](#)
- [extracted_colors_dict](#)
 - [Extractor, 31](#)
- [extraction_utils.py, 37](#)
- [Extractor, 27](#)
 - [__init__, 28](#)
 - [base_color_dict, 31](#)
 - [check_pastel_conversion, 28](#)
 - [construct_palette_dictionary, 28](#)
 - [construct_scheme_dictionary, 29](#)
 - [convert_pastel, 29](#)
 - [convert_pastel_dark, 29](#)
 - [convert_pastel_light, 31](#)
 - [convert_pastel_normal, 31](#)
 - [extracted_colors_dict, 31](#)
 - [hsv_img_matrix_2d, 32](#)
 - [output_filepath, 32](#)
 - [palette_dict, 32](#)
 - [pastel_dark, 32](#)
 - [pastel_light, 32](#)
 - [pastel_normal, 32](#)
 - [ratio_dict, 32](#)
 - [run, 31](#)
- [Extractor.py, 39](#)
- [EXTRACTORS](#)
 - [pypalex.__main__, 6](#)
- [file_utils.py, 39](#)
- [FILENAMES](#)
 - [pypalex.__main__, 6](#)
- [find_closest_to_centroid](#)
 - [pypalex.extraction_utils, 19](#)

- generate_background_and_foreground
 - pypalex.extraction_utils, 19
- generate_black_and_white
 - pypalex.extraction_utils, 20
- generate_panes
 - pypalex.print_utils, 25
- generate_remaining_colors
 - pypalex.extraction_utils, 20
- get_ansi_color_codes
 - pypalex.print_utils, 25
- get_color_escape
 - pypalex.print_utils, 25
- get_dominant_hue
 - pypalex.extraction_utils, 20
- get_left_and_right_colors
 - pypalex.extraction_utils, 21
- get_rgb_palette
 - pypalex.print_utils, 26
- GREEN_HEX
 - pypalex.constants, 10
- GREEN_HUE
 - pypalex.constants, 10
- GREEN_HUE_RANGE
 - pypalex.constants, 10
- GREEN_RGB
 - pypalex.constants, 10
- handle_args
 - pypalex.__main__, 5
- hex_to_rgb
 - pypalex.conversion_utils, 13
- hsv_img_matrix_2d
 - Extractor, 32
- hsv_to_hex
 - pypalex.conversion_utils, 13
- hsv_to_rgb
 - pypalex.conversion_utils, 14
- image_utils.py, 40
- LIGHT_BRIGHTNESS_RANGE
 - pypalex.constants, 11
- MAGENTA_HEX
 - pypalex.constants, 11
- MAGENTA_HUE
 - pypalex.constants, 11
- MAGENTA_HUE_RANGE
 - pypalex.constants, 11
- MAGENTA_RGB
 - pypalex.constants, 11
- main
 - pypalex.__main__, 5
- no_args_help_message
 - pypalex.arg_messages, 8
- NORM_BRIGHTNESS_RANGE
 - pypalex.constants, 11
- output_filepath
 - Extractor, 32
- OUTPUT_FILEPATHS
 - pypalex.__main__, 6
- OUTPUT_PATH
 - pypalex.__main__, 6
- OUTPUT_TAIL
 - pypalex.__main__, 6
- palette_dict
 - Extractor, 32
- PASTEL_BRIGHTNESS_RANGE
 - pypalex.constants, 11
- PASTEL_D
 - pypalex.__main__, 6
- pastel_dark
 - Extractor, 32
- PASTEL_L
 - pypalex.__main__, 7
- pastel_light
 - Extractor, 32
- PASTEL_N
 - pypalex.__main__, 7
- pastel_normal
 - Extractor, 32
- PASTEL_SATURATION_RANGE
 - pypalex.constants, 11
- print_default_scheme_preview
 - pypalex.print_utils, 26
- print_utils.py, 41
- process_helper
 - pypalex.image_utils, 23
- process_image
 - pypalex.image_utils, 23
- PROPER_IMAGES
 - pypalex.__main__, 7
- pypalex, 3
 - pypalex.__main__, 3
 - check_path, 4
 - check_source, 4
 - check_sources, 5
 - extract_color_palettes, 5
 - EXTRACTORS, 6
 - FILENAMES, 6
 - handle_args, 5
 - main, 5
 - OUTPUT_FILEPATHS, 6
 - OUTPUT_PATH, 6
 - OUTPUT_TAIL, 6
 - PASTEL_D, 6
 - PASTEL_L, 7
 - PASTEL_N, 7
 - PROPER_IMAGES, 7
 - SAVE_CHECK, 7
 - set_global_args, 5
 - setup_argument_parser, 6
 - SHOW_PREVIEW, 7
- pypalex.arg_messages, 7
 - bad_path_message, 8
 - bad_source_message, 8

- no_args_help_message, 8
- pypalex.constants, 8
 - BLACK_BRIGHTNESS_RANGE, 9
 - BLACK_HEX, 9
 - BLACK_RGB, 9
 - BLUE_HEX, 9
 - BLUE_HUE, 9
 - BLUE_HUE_RANGE, 9
 - BLUE_RGB, 10
 - CYAN_HEX, 10
 - CYAN_HUE, 10
 - CYAN_HUE_RANGE, 10
 - CYAN_RGB, 10
 - DARK_BRIGHTNESS_RANGE, 10
 - GREEN_HEX, 10
 - GREEN_HUE, 10
 - GREEN_HUE_RANGE, 10
 - GREEN_RGB, 10
 - LIGHT_BRIGHTNESS_RANGE, 11
 - MAGENTA_HEX, 11
 - MAGENTA_HUE, 11
 - MAGENTA_HUE_RANGE, 11
 - MAGENTA_RGB, 11
 - NORM_BRIGHTNESS_RANGE, 11
 - PASTEL_BRIGHTNESS_RANGE, 11
 - PASTEL_SATURATION_RANGE, 11
 - RED_HEX, 11
 - RED_HUE, 11
 - RED_HUE_RANGE_MAX, 12
 - RED_HUE_RANGE_MIN, 12
 - RED_RGB, 12
 - SATURATION_TOLERANCE_RANGE, 12
 - WHITE_HEX, 12
 - WHITE_RGB, 12
 - YELLOW_HEX, 12
 - YELLOW_HUE, 12
 - YELLOW_HUE_RANGE, 12
 - YELLOW_RGB, 12
- pypalex.conversion_utils, 13
 - hex_to_rgb, 13
 - hsv_to_hex, 13
 - hsv_to_rgb, 14
 - rgb_to_hex, 14
 - rgb_to_hsv, 14
- pypalex.extraction_utils, 15
 - borrow_color, 16
 - calculate_centroid, 16
 - check_missing_color_types, 16
 - check_missing_colors, 17
 - construct_base_color_dictionary, 17
 - extract_color_palettes, 18
 - extract_color_types, 18
 - extract_dominant_color, 18
 - extract_ratios, 19
 - find_closest_to_centroid, 19
 - generate_background_and_foreground, 19
 - generate_black_and_white, 20
 - generate_remaining_colors, 20
 - get_dominant_hue, 20
 - get_left_and_right_colors, 21
 - sort_by_sat_and_bright_value, 21
- pypalex.Extractor, 22
- pypalex.file_utils, 22
 - save_default_scheme_to_file, 22
 - save_palette_to_file, 22
- pypalex.image_utils, 23
 - process_helper, 23
 - process_image, 23
 - rescale_image, 24
- pypalex.print_utils, 24
 - generate_panes, 25
 - get_ansi_color_codes, 25
 - get_color_escape, 25
 - get_rgb_palette, 26
 - print_default_scheme_preview, 26
- PyPalEx: The Python Palette Extractor, 1
- ratio_dict
 - Extractor, 32
- RED_HEX
 - pypalex.constants, 11
- RED_HUE
 - pypalex.constants, 11
- RED_HUE_RANGE_MAX
 - pypalex.constants, 12
- RED_HUE_RANGE_MIN
 - pypalex.constants, 12
- RED_RGB
 - pypalex.constants, 12
- rescale_image
 - pypalex.image_utils, 24
- rgb_to_hex
 - pypalex.conversion_utils, 14
- rgb_to_hsv
 - pypalex.conversion_utils, 14
- run
 - Extractor, 31
- SATURATION_TOLERANCE_RANGE
 - pypalex.constants, 12
- SAVE_CHECK
 - pypalex.__main__, 7
- save_default_scheme_to_file
 - pypalex.file_utils, 22
- save_palette_to_file
 - pypalex.file_utils, 22
- set_global_args
 - pypalex.__main__, 5
- setup_argument_parser
 - pypalex.__main__, 6
- SHOW_PREVIEW
 - pypalex.__main__, 7
- sort_by_sat_and_bright_value
 - pypalex.extraction_utils, 21
- WHITE_HEX
 - pypalex.constants, 12

WHITE_RGB

`pypalex.constants`, [12](#)

YELLOW_HEX

`pypalex.constants`, [12](#)

YELLOW_HUE

`pypalex.constants`, [12](#)

YELLOW_HUE_RANGE

`pypalex.constants`, [12](#)

YELLOW_RGB

`pypalex.constants`, [12](#)